Accessing Entire Operations Symbols

This subsection covers the following topics:

- Handling Entire Operations Symbols
- Description of the Function Codes
- Sequential Reading in a Symbol Table
- Example of Sequential Symbol Table Reading

Handling Entire Operations Symbols

You can handle symbols in symbol tables using the following statement:

```
CALLNAT ’NOPUSY4N’
    FUNCTION RC OWNER NETWORK RUN SYMBOL-TABLE SYMBOL-NAME FORMAT
    VALUE USER TIME
```

Meaning of the parameters:

Parameter	**Format**	**Usage**		
FUNCTION	A01	in	Function code:	
			D	Delete one multiple value.
			M	Add one multiple value.
			N	Test next symbol.
			P	Test next symbol to be prompted.
			R	Reset a symbol.
			S	Set (add or modify) a symbol.
			T	Test existence of a symbol and inquire its value.

RC	N03	out	**Return code:**	
			0	Function ok; symbol found.
			1	Symbol not found.
			2	Invalid numeric value.
			3	Invalid format.
			4	Format modification attempted.
			5	Multiple table full.
			6	Value missing.
			7	Value not found.
			10	Invalid value by user exit.
			11	User exit not found.
			12	Access to user exit not allowed (Natural Security).
			20	OK; is a multiple value.
			30	OK; the master symbol was also modified.
			101	Invalid function code.
			102	Parameter missing.
			111	Symbol table name starts with reserved prefix =EOR=.
OWNER	A10	in	Owner of the symbol table.	
NETWORK	A10	in	Network (for active symbol table only).	
RUN	P13	in	Run (for active symbol table only).	
SYMBOL-TABLE	A10	in	The symbol table.	
SYMBOL-NAME	A20	in	The symbol name.	
		out	(for function codes **N** and **P**)	
FORMAT	A01	in	Symbol format.	
		out	(for function codes **N**, **P** and **T**)	
			" " or A	Alphanumeric. No case conversion.
			D	Date in the format: YYYYMMDD
			L	Alphanumeric. Conversion to lower case.
			N	Numeric.
			U	Alphanumeric. Conversion to upper case.
VALUE	A80	in	Symbol value.	
		out	(for function codes **N**, **P** and **T**)	
USER	A08	out	User who made the last modification.	
TIME	T	out	Time of the last modification.	

Description of the Function Codes

Meaning of the function codes:

Code	Description
D	**Delete** one multiple value. Resets one value in a multiple-value symbol. If this is the last symbol value, the whole symbol is removed from the table.
M	**Add** one **multiple** value. Sets one more value in a multiple-value symbol. **Note:** Even if the quantity of the multiple values is 1, the single value remains a multiple value. It will not be converted into a standard value.
N	**Test** next symbol. Tries to find the next symbol (in alphabetic order) from the specified symbol. The given name is overwritten by the name found. The other fields are returned as in function **T**. To find the first symbol of a symbol table, the symbol name can be omitted. If the end of the symbol table is reached, code **1** is returned.
P	**Test** next symbol to be prompted. Finds the next symbol in alphabetical order to be prompted for the current network/job run. Otherwise works like function code **N**.
R	**Reset** a symbol. The symbol is removed from the symbol table.
S	**Set** a symbol. If the symbol does not exist in the symbol table, it is inserted; if it already exists, it is overwritten.
T	**Test** a symbol. If the symbol is not found, code **1** is returned. If the symbol exists, its format and value are returned. The fields User and Time contain the user and time stamp of the last modification.

Notes:

1. To access a **master** symbol table, the fields NETWORK and RUN must be empty. To access an **active** symbol table, the fields NETWORK and RUN must be supplied by the caller.
2. If a plausibility check user exit is defined for the symbol, it is invoked from this API routine, too. Make sure that the user exit is accessible in the calling environment. The library containing the user exit must be defined as STEPLIB for the executing environment of these APIs. A symbol value is **rejected** if the exit returns **not ok** (RC=10) or if the exit is missing (RC=11).

Sequential Reading in a Symbol Table

To read symbols sequentially from a master or active symbol table, proceed as follows:

1. Set function to **N**.
2. If you want to start reading at the start of the table, reset SYMBOL-NAME first; otherwise put a start name into SYMBOL-NAME.
3. Call the user routine in a REPEAT loop; leave it, if RC is not **0** and not **20**.
4. Leave SYMBOL-NAME unchanged as start value for the next call.
5. Reset SYMBOL-VALUE before the next call, if RC=0.
6. Do not reset SYMBOL-VALUE, if RC=20.

Example of Sequential Symbol Table Reading

```
MOVE ’N’ TO FUNCTION
RESET SYMBOL-NAME
R1.  REPEAT
     CALLNAT ’NOPUSY4N’ RC  ...

     DECIDE ON FIRST VALUE OF RC
     VALUE 0, 20 IGNORE
     VALUE 1 ESCAPE BOTTOM (R1.)
     NONE VALUE
            /* Error handling
     END-DECIDE
            /* process symbol here
     IF RC NE 20
          RESET VALUE
     END-IF
END-REPEAT
```